СТРАТЕГИЯ УСТАНОВЛЕНИЯ ВЫВОДИМОСТИ ФОРМУЛ В СТРУКТУРНЫХ ФУНКЦИОНАЛЬНЫХ МОДЕЛЯХ

Д.А. Коваленко, В.Б. Новосельцев

Томский политехнический университет
E-mail: msmaklaud@gmail.com

Рассматривается исчисление рекурсивных предложений для теории структурных функциональных моделей. Исследуются вопросы разрешимости и полноты исчисления. Предлагаются стратегия и алгоритм установления выводимости формул исчисления, показывается корректность алгоритма и определяется оценка его эффективности.

Введение

Рассматривается формальное исчисление, используемое для установления выводимости предложений в рамках теории структурных (С-) функциональных моделей [1, 2]. Проведенные исследования традиционно относят к области *искусственного интеллекта*. Прикладной стороной исследования является создание интеллектуальных программных комплексов широкого класса, опирающихся на дедуктивный подход в области менеджмента знаний. Термин «знание» здесь имеет ограниченную трактовку: *знания* представляются специальными строго определенными информационными структурами, допускающими автоматизированную обработку с использованием аппарата формальной логики. Представляемые результаты могут быть использованы при реализации различных информационных комплексов, – от экспертных [3] и когнитивных систем [4] до инструментальных и CASE-оболочек быстрого прототипирования программ [5].

В первом разделе определяются используемые понятия и необходимые соглашения. Во втором вводится понятие интерпретации С-модели и формулируется исчисление функциональных предложений для теории структурных моделей [6], а также исследуются некоторые свойства построенного исчисления. Третий раздел посвящен описанию стратегии и базового алгоритма установления выводимости предложений языка. Для алгоритма доказывается свойство корректности, и устанавливаются сложностные характеристики. Наконец, в последнем разделе кратко рассмотрены особенности введения рекурсии.

1. Базовые определения

Дадим основные определения. Прежде всего, зафиксируем сигнатуру Σ.

Определение 1. Зафиксируем $\Sigma=(A,F,P,D)$, где A, F, P и D – не более чем счетные множества (элементарных) имен объектов, функциональных символов, селекторных символов и имен схем, соответственно. Выделим в D непустое конечное подмножество $E\subset D$ имен *примитивных* или *первичных* схем.

Элементы множества A используются для формирования (*имен*) *объектов*. Связь объекта a с некоторой схемой S отражается записью $S(a)$. Если объект a связан со схемой S и $S\in E$, стандартная запись $S(a)$ заменяется записью a^S либо a, когда ссылка на схему не важна или очевидна из контекста.

Определение 2. Выражение вида

$$f{:}a_1,\dots,a_n\to a_0, \qquad (1)$$

где $a_i\in A$, $i=0,\dots,n$ – имена объектов, называется функциональной связью (ФС). В записи (1) $f\in F$ – это имя ФС, a_i – ее аргументы ($i = 1,\dots,n$), a_0 – результат.

Имена объектов при моделировании прикладной предметной области (ПО) формируются следующим образом.

Определение 3. Пусть $a\in A,\sigma$ – имя объекта, тогда a, $\sigma.a$ и $a.\sigma$ также являются именами. В записи $\sigma.a$ σ называется префиксом. *Длина* имени σ определяется числом вхождений элементов A (с учетом возможных повторений).

Функциональные связи используются при задании (непервичных) схем объектов моделируемой ПО.

Определение 4. Схема *S* объекта *r* определяется выражением вида

$$S(r) = r.(S_{01}(a_{01}),..., S_{0N_0}(a_{0N_0}),$$
$$if \quad p_1 \supset S_{11}(a_{11}),..., S_{1N_1}(a_{1N_1})□...□ \quad (2)$$
$$p_k \supset S_{k1}(a_{k1}),..., S_{kN_k}(a_{kN_k}) \;\; fi \mid filter),$$

где $S \in D \backslash E$, $r \in A$. Для всех возможных значений индексов *i, j*, $S_{ij} \in D$ – *собственные подсхемы* схемы *S*, $a_{ij} \in A$ – ее *собственные величины*, $p_i \in P$ – выбирающие *селекторы*. В правой части (2) *r* называется *префиксом* схемы, участок до вертикальной черты – *заголовком*, фрагмент *if…fi – вариантной* его *частью*, а остальная часть заголовка – *постоянной частью*. Иероглиф *filter* скрывает список *собственных* ФС схемы *S*. Префикс *r* может «проноситься» в скобочный фрагмент, так что $r.(S(a),...) \rightleftharpoons (r.S(a),...) \rightleftharpoons (S(r.a),...)$. Аналогичным образом префиксируются имена селекторов и отображений, а также имена величин, вовлеченных в формирование ФС из *filter*.

Селекторы на интерпретации I получают истинностные (шкальные) значения и образуют полную систему, т. е. $\forall i,j,\ i \neq j,\ p_i|_I \& p_j|_I =$Л и $p_1|_I \vee ... \vee p_k|_I =$И.

Определение 5. Пусть $S(r)=(...S_{ij}(r.a_{ij})...)$ – схема. Если $S_{ij} \in E$, то $r.a_{ij}$ – имя величины схемы *S*. Если $S_{ij} \in D \backslash E$ и α – имя величины схемы S_{ij}, то $r.a_{ij}.\alpha$ – имя величины схемы *S*. Величины схемы *S* будем также называть её атрибутами.

Определение 6. Рассмотрим схему *S*. ФС $f{:}a_1,...,a_n \to a_0$ из *filter* называется допустимой для схемы *S*, если и только если $a_0, a_1,...,a_n$ – атрибуты схемы *S*. Схема называется синтаксически правильной, если *filter* содержит только допустимые ФС.

В дальнейшем рассматриваются только синтаксически правильные схемы. Помимо этого считается, что ФС из набора *filter*, отвечают следующим требованиям: (а) ФС содержит по крайней мере один *собственный* атрибут схемы; (б) в наборе *filter* нет ФС, в которые вовлечены атрибуты из разных ветвей вариантной части схемы *S* или атрибуты из вариантных частей её *подсхем*; (в) длина любого атрибута (имени), входящего в ФС, не превышает двух.

С каждым атрибутом связывается условие допустимости его существования в схеме.

Определение 7. Факт наличия условия *p* у атрибута *a* отмечается записью *a*/*p*, при этом *a*/*p* называется *у-атрибутом*. Считается, что с атрибутами, определёнными в общей части заголовка схемы, связывается условие И, – в этом случае фрагмент «/*p*» опускается.

Определение 8. Структурной (С-) моделью называется конечная совокупность (описаний) схем $M=(S_1,...,S_m)$, где каждая S_i, $i=1,...,n$, является элементарной или задана в соответствие с определением 4. С-модель *M* является (синтаксически) замкнутой, если и только если для каждого её элемента $S_i \in M$ схемы, встречающиеся в определении S_i, являются элементарными либо определены в описании С-модели *M*.

Описывая С-модель для предметной области, мы строим специальную (формальную) теорию, в которой можно решать, в частности, задачу установления выводимости предложений соответствующего языка и генерации реализующей вывод программы.

Постановка задачи в С-модели носит непроцедурный характер – в ней указываются лишь исходные и искомые атрибуты некоторой схемы. Содержательно, постановка задачи на модели есть задание на построение схемы алгоритма, реализующего требуемые вычисления. Схема алгоритма извлекается из (конструктивного) доказательства соответствующей теоремы существования (в силе [7]). Схема становится алгоритмом (программой) в результате задания интерпретации модели.

Определение 9. Задачей в С-модели *M* называется тройка $T=(A_0,X_0,S)$, где A_0 и X_0 – наборы имён, соответственно, исходных и искомых величин, а *S* – схема С-модели *M*, в которой определены эти имена.

При исследовании свойств С-моделей и алгоритмов установления выводимости в них потребуется понятие развёртки схемы *S*.

Определение 10. Пусть $S(r)=r.(...S_{ij}(a_{ij}),...|\,filter)$ – схема, и $S_{ij} \notin E$. Развёрткой *S* на атрибуте a_{ij} называется выражение, получающееся в результате: (а) подстановки в заголовок исходной схемы на место $S_{ij}(a_{ij})$ заголовка схемы S_{ij} и (б) присоединения к набору *filter* схемы *S* ФС схемы S_{ij}. Все добавленные в схему *S* в результате такого действия атрибуты, имена селекторов и отображений модифицируются префиксом $r.a_{ij}$. Под полной развёрткой *S* С-модели понимается объект, полученный из схемы *S*, у которой в результате последовательности развёрток на атрибутах в заголовке остаются только атрибуты, связанные с элементарными схемами.

Понятие схемы нетрудно переопределить с тем, чтобы оно оставалось корректным для развёртки на атрибуте и полной развёртки. При построении С-модели допускаются возможность рекурсивных определений схем. В общем случае рекурсивная конструкция должна содержать, по меньшей мере, одну цепочку определений вида $S_1=(...[S_2]...)$, $S_2=(...[S_3]...),...,S_k=(...[S_1]...)$, обеспечивающую участие в дефиниции некоторого понятия ссылки на себя. Ясно, что для рекурсивной схемы полная развертка не определенна.

2. Интерпретация С-модели и исчисление *SR*

Интерпретация (семантика) С-модели вводится стандартным образом.

Определение 11. Интерпретация I С-модели *M* задаёт: (а) для каждой элементарной схемы $S \in E$ непустое множество (первичный тип) $S|_I$; (б) для каждого функционального символа $f \in F$ – отображение $f|_I{:}S_1|_I \times ... \times S_n|_I \to S_0|_I$; (в) для каждого селектора $p \in P$ – булевское отображение $p\,|_I{:}S_1|_I \times ... \times S_n|_I \to \{$И,Л$\}$.

В результате задания интерпретации каждой неэлементарной схеме сопоставляется отношение, т. е. множество наборов, удовлетворяющих некоторым условиям. Наборы принадлежат декартову произведению множеств, сопоставленных подсхемам схемы S. Элементы наборов поименованы атрибутами заголовка. Если в заголовке схемы присутствует альтернативная часть, то наборы отношения выбираются из размеченного декартова произведения [8]. Разметку определяют селекторы альтернативной части. Термин «тип атрибута» служит сокращением для выражения «множество, сопоставленное схеме, с которым связан атрибут». В отличие от *первичного* типа такой тип мы будем называть (*непервичным* или просто) *типом*, наборы типа – *кортежами*, а их именованные элементы – *компонентами*.

Отображения, которые сопоставляются ФС схемы в результате задания семантики С-модели, определяют соотношения между компонентами кортежей. Выполнение этих соотношений указывает *принадлежность* кортежа типу. С компонентами кортежей связываются реализации условий, которые приписаны соответствующим этим компонентам y-атрибутам.

Для удобства и большей компактности рассуждений вводится ряд дополнительных соглашений, не имеющих принципиального характера:

Используется только *двухальтернативная* форма вариантной части в (2): $if\ p \supset ...;...fi$.

Вводится понятие *предложения вычислимости* (ПВ) вида $F{:}A \rightarrow X$, служащее сокращением для $\{F_{xi}{:}A \rightarrow x_i\}$, где F_{xi} – определяемые ниже программные термы, формирующие F, а $X \rightleftharpoons \cup\{x_i\}$.

Определение 12. Рассмотрим С-модель $M=(...S...)$. Пусть $a_1/p_1,...,a_n/p_n,\ x_1/q_1,...,x_m/q_m$ – y-атрибуты развёртки S, $\Psi \rightleftharpoons F{:}a_1/p_1,...,a_n/p_n \rightarrow x_1/q_1,...,x_m/q_m$ – предложение вычислимости, а r – кортеж, определяемый заголовком развёртки при некоторой интерпретации I. Будем говорить, что $\Psi|_I$ имеет смысл для кортежа r, если $p_i|_I$=И, $i=1,...,n$ и $q_j|_I$=И, $j=1,...,m$.

Определение 13. Будем говорить, что ПВ $\Psi \rightleftharpoons F{:}A \rightarrow X$ *удовлетворяет* схеме S, если при любой интерпретации I тип $S|_I$ не содержит двух кортежей, для которых $\Psi|_I$ имеет смысл, таких что их компоненты совпадают для всех атрибутов множества A, но не совпадают по крайней мере для одного атрибута множества X. При этом для произвольного кортежа r, который принадлежит отношению $S|_I$, результатом применения $F|_I$ к компонентам r, именованным атрибутами множества A, является набор компонент, именованных атрибутами множества X. Все ФС схемы удовлетворяют ей по определению.

В исчислении имеется три сорта термов.

Определение 14. (1) Термы первого сорта (a-термы) представляют собой упорядоченные списки атрибутов, имена которых строятся из элементов А. (2) Термы второго сорта (y-термы) строятся из термов первого сорта и селекторных символов, принадлежащих множеству Р, согласно следующей рекурсивной процедуре. Пусть $p^{(n)} \in$ Р – селектор, реализуемый n-местной булевской функцией, A – n-элементный терм первого сорта, такой, что его i-й атрибут связан с той же схемой, что и i-й аргумент p; P и Q – термы второго сорта; α – некоторый атрибут; «–» и «&» – символы отрицания и конъюнкции, соответственно. Тогда перечисленные выражения являются термами второго сорта (других термов второго сорта нет): $\alpha.p(A)$, $-P$, $P\&Q$. (3) Наконец, программные (np–) термы третьего сорта строятся из термов всех сортов по следующим правилам. Пусть P – терм второго сорта; f, g, h – функциональные символы (элементы множества F); F_1, F_2, F_3 – термы третьего сорта и имеются вхождения символа g в F_3; α – некоторый атрибут; $F_3[h/g]$ – обозначение замены вхождения g на h в терме F_3. Тогда $\alpha.h{:}... \rightarrow ...$ – терм третьего сорта; $F_1;F_2$ – терм, построенный с применением оператора композиции; $if\ P\ then\ F_1\ else\ F_2\ fi$ – терм (оператор ветвления); $h=if\ P\ then\ F_1\ else\ F_3[h/g]\ fi$ – терм (оператор рекурсии). – Других термов третьего сорта нет.

В рамках введенных соглашений используется базовый набор правил вывода:

(0) *схема аксиом*: $\square N{:}A \rightarrow A$.

(1) $\dfrac{\vdash F{:}A \rightarrow X, Z}{\vdash F{:}A \rightarrow X}$ (*правило сужения*);

(2) $\dfrac{\vdash F{:}A \rightarrow X, Z/P \quad \vdash f{:}Z \rightarrow x/p}{\vdash F;f{:}A, Z/P \rightarrow X, Z, x/P\&p}$ (*правило композиции*),

здесь P – конъюнкция (шкальных) условий достижимости атрибутов из Z, а p – условие допустимости x (условие достижимости – это условие, при котором в процессе построения доказательства обеспечивается достижимость некоторого атрибута, а условие допустимости – это условие, при котором атрибут имеет смысл в описании).

(3) $\dfrac{\vdash F_1{:}A \rightarrow X, x/Q\&p \quad \vdash F_2{:}A \rightarrow X, x/Q\&-p}{\vdash if\ p\ then\ F_1\ else\ F_2\ fi{:}A \rightarrow X, x/Q}$

(*правило ветвления*);

(4) $\vdash F_1{:}A \rightarrow X, x/Q\&p$

$$\frac{g_1{:}\,n_{б_1}^{k_1}(A) \rightarrow n_{б_1}^{k_1}(X),...,g_s{:}\,n_{б_s}^{k_s}(A) \rightarrow n_{б_s}^{k_s}(X) \vdash F_2{:}A \rightarrow X, x/Q\&-p}{h=if\ p\ then\ F_1\ else\ F_2[h/g_1]...[h/g_s]\ fi{:}A \rightarrow X/Q}$$

(*правило рекурсии*).

Построенное исчисление будем называть исчислением SR (*structured recursive calculus*).

Для исчисления SR справедливы следующие утверждения.

Теорема 1. Исчисление SR является корректным относительно понятия ПВ, удовлетворяющего схеме S.

Теорема 2. Исчисление SR является полным относительно понятия ПВ, удовлетворяющего схеме S.

3. Стратегия и алгоритм установления выводимости предложений *SR*

Алгоритм, включающий и стратегию вывода, описывается с использованием некоторого псевдокода, структуры управления которого традиционны для императивных языков программирования. При описании алгоритма используются следующие обозначения:

- C_k – множество *y*-атрибутов (развертки) исходной схемы, достижимость которых установлена на текущий момент;
- *по*-аксиома – *проблемно-ориентированная* или *предметная* аксиома – ФС из текущей схемы;
- «вход в подсхему» – наличие в C_k *y*-атрибутов, которые принадлежат подсхеме текущей схемы;
- A_0 и *S* берутся из формулировки задачи *T*.

Описание алгоритма:

«установить i=0, C_0=A_0»;

«поднять флаг «*продолжать доказательство*»»;

«текущей схемой объявить *S*»;

while «поднят флаг «*продолжать доказательство*»» ***do if*** «имеется *по*-аксиома текущей схемы, аргументы которой входят в C_i» ***then*** «применить правило композиции, и, если возможно, правило ветвления»;

«построить C_{i+1}: к C_i добавить новые *y*-атрибуты либо изменить условия достижимости ранее полученных»;

«установить i=i+1»;

«если определился вход в подсхему, запомнить её в стеке»;

else if «имеется подсхема, в которую определён вход» ***then*** «объявить её текущей схемой»;

else if «текущая – рекурсивная схема» ***then***

«применить правило введения рекурсии; установить i=i+1»;

«определить C_{i+1}»;

else «объявить текущей внешнюю схему»;

fi fi fi;

if «текущая – исходная и нет *по*-аксиом, аргументы которых входят в C_i « ***then*** «опустить флаг «*продолжать доказательство*»» ***fi od***.

Для приведенного алгоритма справедлива следующая теорема.

Теорема 3. Приведённый алгоритм корректен, т. е. обеспечивает нахождение всех выводимых атрибутов. Верхней оценкой сложности алгоритма является выражение $O(|M|^3)$.

4. Особенности введения рекурсии

Для того чтобы рассмотреть особенности введения правила рекурсии дадим определение замыкания.

Определение 15. Пусть $S(r) = r.(A,X, \ldots \mid filter)$ и пусть *A* – некоторый набор атрибутов схемы *S*. Замыканием A^+ для *A* относительно *S* будем называть объединение всех таких атрибутов *X*, что ПВ $F{:}A \rightarrow X$ может быть получено с использованием правил SR из информации о схеме *S*.

Теперь предположим, что мы имеем рекурсивную детерминированную схему $S(\alpha)=\alpha.(A,\ldots if\ p \supset \ldots;\ S(v),\ldots\ fi \mid filter)$. Пусть, для простоты, все подсхемы за исключением *S* являются элементарными. По сделанному предположению схема *S* стала текущей с «входом», определяемым множеством *A*.

Введем следующие обозначения Arg и Res – множество аргументов и результатов рекурсивного ПВ соответственно;

Опишем на содержательном уровне процесс, который позволяет определить Arg и Res из заключения правила введения рекурсии. Сначала, не применяя правила (4) исчисления SR для схемы *S*, выводим из атрибутов *A* всё, что возможно. В Arg включаем все входные атрибуты *A*. Для этих атрибутов показана достижимость при условии, которое приписано постоянной части схемы *S* (условие «входа» в *S*). После этого определяем замыкание A^+ и объявляем его окончательным множеством аргументов Arg.

В Res помешаем *y*-атрибуты из постоянной части схемы *S* не попадающие в Arg, для которых показана достижимость от Arg при условии нерекурсивной ветви, и такие, что вовлечены в *левые* части ФС вида $\alpha.f{:}\ldots,\alpha.v.b,\ldots \rightarrow \alpha.c$ – «выводящие» из рекурсивного вхождения схемы $\alpha.S(v)$. Наконец, транзитивное замыкание полученного множества атрибутов Res^+ объявляем окончательным множеством результатов рекурсивного ПВ.

Замечание 1. Если любое из транзитивных замыканий окажется пустым, применение правила введения рекурсии оказывается невозможным.

Замечание 2. Если транзитивное замыкание строить без учета дополнительных соображений, затраты времени могут стать экспоненциальными по числу атрибутов.

Заключение

Как показывает практика, стратегия, обеспечиваемая теорией С-моделей в комплексе с исчислением SR допускает весьма эффективные реализации разнообразных когнитивных систем. В силу того, что задача установления выводимости тесно связана с нахождением транзитивных замыканий специальных отношений, естественные варианты реализации опираются на прекрасно зарекомендовавшие себя и весьма эффективные подходы. Прежде всего, среди последних имеет смысл выделить алгоритмы Армстронга и Диковского[10]. В разных модификациях эти алгоритмы обладают полиномиальными оценками с невысокими степенями (линейные либо квадратичные), хотя и связаны с существенно более бедными по сравнению с

описываемыми здесь моделями. В первом случае речь идет о связях на атрибутах реляционных схем в базах данных, а во втором – рассматриваются линейные вычислительные модели.

СПИСОК ЛИТЕРАТУРЫ

1. Минц Г.Е. Резолютивные исчисления для неклассических логик. // 9-ый Советский Кибернетический симпозиум. – 1981. – Т. 2. – № 4. – С. 34–36.
2. Новосельцев В.Б. Использование методов ИИ в системах автоматизации атмосферно-оптических исследований. // Журнал оптики атмосферы и океана. Спец. выпуск. – 1998.- № 2. – С. 56–63.
3. Alty J.L. Expert systems. – Washington: NCC Publications, 1987. – 190 p.
4. Поспелов Д.А. Логико-лингвистические модели в системах управления. – М.: Энергоиздат, 1981. – 231 с.
5. Кахро М.И., Калья А.П., Тыугу Э.Х. Инструментальная система программирования ЕС ЭВМ (ПРИЗ). – М.: Финансы и статистика, 1981. – 216 с.
6. Новосельцев В.Б. Синтез рекурсивных программ в системе СПОРА // Институт теоретической астрономии АН СССР. Алгоритмы небесной механики. – 1985. – № 43. – С. 21–45.
7. Минц Г.Е., Тыугу Э.Х. Полнота правил структурного синтеза // Доклады АН СССР. – 1982. – Т. 2. – № 6. – С. 41–60.
8. Дал У., Дейкстра Э., Хоар К. Структурное программирование. – М.: Мир, 1975. – 327 с.
9. Ульман Дж. Основы систем баз данных. – М.: Финансы и статистика, 1983. – 432 с.
10. Диковский А.Я. Детерминированные вычислительные модели // Техническая кибернетика. – 1984. – Т. 5. – № 5. – С. 84–105.

УДК 004.89